

# Penalized Splines for Smooth Representation of High-dimensional Monte Carlo Datasets

Nathan Whitehorn<sup>a,\*</sup>, Jakob van Santen<sup>a</sup>, Sven Lafebre<sup>b</sup>

<sup>a</sup>University of Wisconsin - Madison, Dept. of Physics, Madison WI, 53705 USA

<sup>b</sup>Pennsylvania State University, Dept. of Physics, University Park PA, 16802 USA

---

## Abstract

Detector response to a high-energy physics process is often estimated by Monte Carlo simulation. For purposes of data analysis, the results of this simulation are typically stored in large multi-dimensional histograms, which can quickly become both too large to easily store and manipulate and numerically problematic due to unfilled bins or interpolation artifacts. We describe here an application of the penalized spline technique [1] to efficiently compute B-spline representations of such tables and discuss aspects of the resulting B-spline fits that simplify many common tasks in handling tabulated Monte Carlo data in high-energy physics analysis, in particular their use in maximum-likelihood fitting.

**Keywords:** Splines, Monte Carlo, Histograms, Maximum Likelihood

---

## 1. Introduction

Monte Carlo simulation is a common technique for estimating detector response in high-energy physics and tabulated forms of Monte Carlo simulation are commonly used in data analysis as probability density functions in maximum-likelihood fits. As the number of tabulated dimensions increases, numerical problems begin to occur: each bin of the histogram becomes less well filled, statistical fluctuations become larger, zero bins appear. Use of such functions as probability densities with numerical minimization algorithms in maximum likelihood fits can then cause substantial problems with convergence and makes parameter extraction using this approach difficult. The use of penalized spline fits [1] in place of raw histograms, however, allows the resolution of most of these problems without resorting to extremely coarse binning and introduces additional capabilities, such as analytic convolution, derivatives, and integrals that can improve fit quality even beyond the removal of numerical instabilities.

The application described here arose in the context of the IceCube detector [2, 3], which uses tabulated Monte Carlo data, among other purposes, for describing photon propagation in glacial ice for simulation of neutrino events, energy reconstruction of muons [4], and for direction, position, and energy reconstruction of electromagnetic and hadronic showers produced by  $\nu_e$  and neutral current neutrino interactions [5]. The heterogeneity of the glacial ice [6] results in a complicated dependence of light propagation on direction, source depth, and receiver depth, resulting in 6-dimensional tables required to describe the light

response of the detector to arbitrarily oriented and positioned showers. Similar problems arise in parametrizing detector response for purposes of event reconstruction in many high-energy detectors, either because of variations in the detector medium (as with IceCube) or because of varied instrumentation technology (as in many collider detectors).

A raw table-based approach caused difficulties, especially for directional reconstruction of showers: the tables can require immense amounts of memory (terabytes) and tabulation can introduce numerical artifacts that resulted in seams in the resulting likelihood space and prevented minimizer convergence. These numerical artifacts arise from averaging over a table cell during binning, from inaccurate interpolation between cells, and from statistical fluctuations as the bins are filled. Any attempt to reduce the size of the tables exacerbated the binning artifacts, and vice versa. An interpolating function was the only viable solution to these problems.

An ideal interpolating function for these types of histograms would have several properties: it would be smooth, fast to evaluate, possible to fit deterministically in a small amount of time, extend easily to large numbers of dimensions, and have parameters that are easy to store, while minimizing introduction of bias and artifacts into the fit surfaces. Basis splines (B-splines) fit this description well: a spline of order  $n$  has  $n - 1$  continuous derivatives, the functions can be evaluated quickly, and fitting them to a histogram is a linear problem, so it can be done quickly with standard techniques. The underlying linear problem is inherently sparse, and can be computed efficiently even for very large histograms using Generalized Linear Array Models (GLAM [7]). The linear formulation can also be extended easily to include a regularization term [1] that

---

\*Corresponding author. Tel: +1 608 265 4978

penalizes erratic sweeps in the fit surface in the absence of strong evidence from the data.

In addition to the general properties listed above, our problem had a number of special requirements that we were able to satisfy through a combination of extensions to the GLAM/penalized-spline fitting technique and optimized routines for the evaluation of tensor-product B-spline surfaces. First, our photon propagation tables were large enough (hundreds of gigabytes) to make it impractical to load the entire 6-dimensional histogram into memory at once, required to perform any fit, so we implemented a method of “stacking” spline surfaces fit to 4-dimensional slices of the full histogram to form a full 6-dimensional interpolating spline surface. Second, we needed to assume the same tables for both simulation and reconstruction of events, which have somewhat different required information. In order to simulate events, we needed to be able to sample from the distribution of photon detection times at fixed points in space using its time-differential form, while for maximum-likelihood reconstruction we needed to be able to integrate the distribution of photon detection times over arbitrary intervals as well as efficiently evaluate the gradients of those integrals in all 6 dimensions. We accomplished all three with a single spline surface by fitting the cumulative distribution of photon arrival times, using the technique of non-negative least-squares to enforce monotonicity, and using a basis of B-splines and their derivatives to simultaneously evaluate both the value and gradient of the resulting surface at little additional computational cost. Third, in the maximum-likelihood reconstruction we needed to be able to account for a family of detector effects and event hypothesis uncertainties that can be approximated by a convolution of the photon detection time distribution with Gaussians of various widths. In order to effect such convolutions without repeating the fit once for each kernel, we implemented a method for quickly computing the coefficients of the convolution of the fit surface with an arbitrary B-spline kernel [8].

We have implemented the spline-fitting techniques mentioned above in a library written in C using sparse matrix operations provided by the SuiteSparse [9] and GotoBLAS 2 [10] libraries, as well as a lightweight library of functions for evaluating the resulting surfaces (code available in the CPC program library). Here we will describe the underlying mathematics of these methods as well as the details of our implementation.

## 2. B-spline surfaces

To create a smooth approximating function in one dimension, we take a linear combination of B-spline functions defined on a common knot vector  $\vec{k}$ , such that the  $n$ th B-spline of order  $m$ ,  $B_m^n$ , is defined on the subset of  $\vec{k}$  from  $\vec{k}_n$  to  $\vec{k}_{n+m+1}$ .

Functions defined in this way have several attractive properties. Because they are linear combinations of smooth

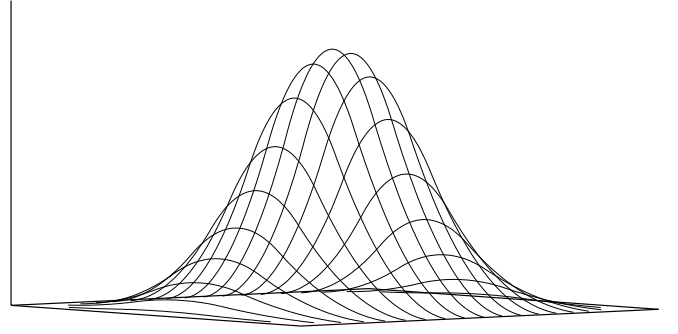


Figure 1: A 2-dimensional tensor product spline constructed from two one-dimensional basis splines.

functions, they are likewise smooth, the amount of information required to represent the function is small, and they can be fit to data using standard linear techniques such as Singular Value, QR, or Cholesky decomposition. In addition, because all but  $m + 1$  of the basis functions are identically zero at any particular point (they have local support), to evaluate the approximation function we need only evaluate the de Boor algorithm [11]  $m + 1$  times. This vastly improves computation time for lookups compared to functional representations like Fourier decompositions or polynomial interpolation that require evaluations of basis functions at every point.

We can extend this approach easily into multiple dimensions by adopting the concept of the tensor product spline. Instead of a knot vector, the tensor product basis functions are defined on a rectangular  $n$ -dimensional knot grid  $\vec{k}$  formed by taking the tensor product of  $n$  one-dimensional knot vectors. Each basis function is then a product of one-dimensional B-splines (Figure 1), which are taken in linear combination as in the one-dimensional case:

$$B(\vec{x}) = \vec{\alpha} \cdot \vec{B} = \vec{\alpha} \cdot \left( B_1^i(\vec{x}_1) \otimes B_2^j(\vec{x}_2) \otimes \dots \right) \quad (1)$$

Because most components of the basis tensor  $\vec{B}$  are zero, as in the one-dimensional case, we recover the same local support property, albeit with more contributing basis functions due to the increased dimensionality of the problem.

## 3. Determining B-spline coefficients

### 3.1. Ordinary least squares

The coefficients of the B-spline basis functions can be found by the usual least-squares formulation:

$$B^\top Bx = B^\top y \quad (2)$$

where  $x$  is the vector of B-spline coefficients,  $y$  the data points in the table, and  $B$  a matrix with one column for

each basis spline; the rows of  $B$  are formed by evaluating the spline at each data point in  $y$ . The value of  $x$  can be found by standard algebraic techniques such as Cholesky decomposition, but the possibility for overfitting and ringing can cause substantial difficulties when using the resulting fits.

### 3.2. Penalized Least Squares

Ringing in the fit surface can be reduced substantially by giving the curve some intrinsic rigidity to follow the general trend instead of following the fluctuations. Using Tikonoff regularization with splines [1] allows us to accomplish this. This is implemented in the context of least squares fitting by adding a term to the equations. Instead of solving Equation 2, we solve the penalized equation:

$$(B^T B + \lambda P^T P) \alpha = B^T y \quad (3)$$

The solution to this equation now minimizes  $\|(B\alpha - y)\| + \lambda\|P\alpha\|$ .  $\lambda$  is then the strength of the regularization (called the smoothing parameter hereafter). For  $\lambda = 0$  we recover the unpenalized normal equations (2). For  $\lambda \rightarrow \infty$ , the regularization term dominates, and we end up with a curve of the shape specified by the regularization term, which is a form of Bayesian prior (for order-2 Tikonoff regularization, this will be a straight line). Details of the form of this matrix can be found in [1].

### 3.3. Penalized Least Squares in Multiple Dimensions

In principle, the extension of least squares fitting to multiple dimensions is easy. As before, we have a set of ( $n$ -dimensional) basis functions  $B$ , to be fit at a set of  $n$ -dimensional positions to a measured set of data  $y$ . With some smoothness constraint on each dimension, we then want to minimize

$$\|B\alpha - y\| + \lambda_1\|P_1\alpha\| + \lambda_2\|P_2\alpha\| + \dots + \lambda_n\|P_n\alpha\| \quad (4)$$

In analogy to Equation 3, we can write down the corresponding system of linear equations:

$$\left( B^T B + \sum_i^n \lambda_i P_i^T P_i \right) \alpha = B^T y \quad (5)$$

The difficulty here arises that  $B$  can be infeasibly large in the multi-dimensional case. For 25 knots on each of 4 axes, and a table with 200 million cells (typical numbers for a single IceCube photon propagation table),  $B$  will have dimension 390625 by 200 million and require 568 TB of RAM.

There are two important things to notice: the first is that  $B$  is quite sparse. Because of the local support property of B-splines, most of the basis functions have no support at most of the data points. As such, we can eliminate all of the zeroes from the matrix using sparse-matrix routines. This reduces memory consumption in the above problem by a factor of 100.

The second thing is that  $B$  is not present in isolation in Equation 5, but instead only occurs as the combination  $B^T B$ . Whereas  $B$  has dimensions of the number of basis functions by the number of data points,  $B^T B$  is a square matrix of side length the number of basis functions. For the example problem above, being able to work directly with  $B^T B$  without the need to ever explicitly form  $B$  would reduce memory consumption by another factor of 500. This also means, critically, that neither the memory requirements nor the CPU requirements of this fitting algorithm would depend on the number of data points, but instead only on the number of basis functions. Working directly with  $B^T B$  can be accomplished using GLAM [7].

### 3.4. Table Stacking and Pseudointerpolation

For each run of the IceCube photon simulation [12] we intend to fit, we produce 4-dimensional light distributions for a single source configuration. For a table-based simulation, many source configurations are simulated and the distributions for an arbitrary source produced from interpolation between the tables for similar sources. For spline tables, we wish to be able to do this as well. The easiest solution, to do a global fit, is impracticable even using GLAM [7]. In addition, because there is no bin averaging between sources, smoothing (and thus least-squares fitting) is unnecessary, and we can stick to interpolation algorithms.

What we do then is to “stack” the individual 4-dimensional tensor-product spline surfaces into a single 6-dimensional tensor-product spline surface. The normalization of the spline basis functions is such that we can simply stack the fit coefficients of each sub-table while creating a knot vector such that the original values of the sub-tables form control points for the interpolating splines in the extra dimensions.

Given a knot vector  $\vec{k}$ , the maxima of the B-splines defined on that vector can be found by solving the equation for the first derivatives of each spline [11, Chapter X, Equation 12]. Given knots spaced at equal intervals  $\Delta\vec{k}$ , these also are the centers of each basis function and can be found at:

$$\vec{B}_n^{max} = \vec{k} - \frac{n-1}{2} \Delta\vec{k} \quad (6)$$

By inverting Equation 6, we can acquire a formula for a knot vector given a vector of centers  $\vec{x}$ :

$$\vec{k} = \vec{x} + \frac{n-1}{2} \Delta\vec{x} \quad (7)$$

We must also add points to the knot vector, since a spline function of order  $n$  made of  $m$  basis functions (and so  $m$  coefficients) must be defined by  $m+n+1$  knots. Thus we need  $n+1$  extra knots, of which the first  $n$  are placed at linearly extrapolated positions before the positions from Equation 7, and one placed at a linearly extrapolated position after the last element.

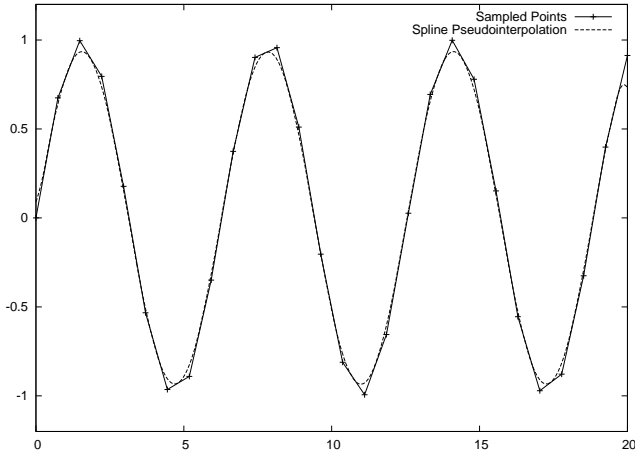


Figure 2: Pseudointerpolation of a set of points sampled from  $\sin(x)$  using the same algorithm as for table stacking. Since B-splines do not interpolate their control points, but instead maximally fill the convex hull formed by those points, the interpolation algorithm will fall slightly short of extrema, while generally maintaining accuracy.

Iterative application of Equation 7 with stacking of coefficients will then produce a new tensor product spline surface with the sub-table locations as control points. It is worth noting that this surface does not exactly interpolate the original tables, as B-splines do not interpolate their control points. Instead, it forms a curve that maximally fills the convex hull of the set of control points (Figure 2). In general, this is an excellent approximation to interpolation, but sharp extrema will be systematically underestimated by a small amount.

#### 4. Cumulative B-spline surfaces

Using a single B-spline surface to represent the distribution of photon arrival times for both simulation and reconstruction purposes posed a special challenge. To be able to efficiently evaluate the integral of the distribution over arbitrary time windows, we fit the cumulative distribution to a tensor-product B-spline surface, using a modified version of the penalized-spline formulation [1] to enforce the monotonicity of the surface. In order to recover the differential distribution and other elements of the gradient from the cumulative form, we implemented an efficient method of simultaneously evaluating the value and gradient of a tensor-product spline surface. Lastly, we implemented a method of quickly – and analytically – convolving a spline surface with a spline kernel in order to approximate detector effects such as PMT transit-time spread during reconstruction.

##### 4.1. Enforcing monotonicity

While the B-spline surface determined by the least-squares condition is by definition the closest fit to the data, it does not necessarily share some desirable properties with the data. In particular, when the data can't be

exactly represented as a spline surface, the fit surface may ring (Figure 3). This can cause the fit surface to become negative in a region where the data are strictly positive or non-monotonic where the data are strictly monotonic. Although this behavior can be substantially reduced by regularization, it is critical that at least monotonicity be exactly obeyed for our application, where we fit for the cumulative photon arrival time distribution, which must by definition be everywhere monotonic. It is possible, however, to express the monotonicity constraint along a single dimension (time, in our case) without substantial modification to the usual least-squares fitting procedure.

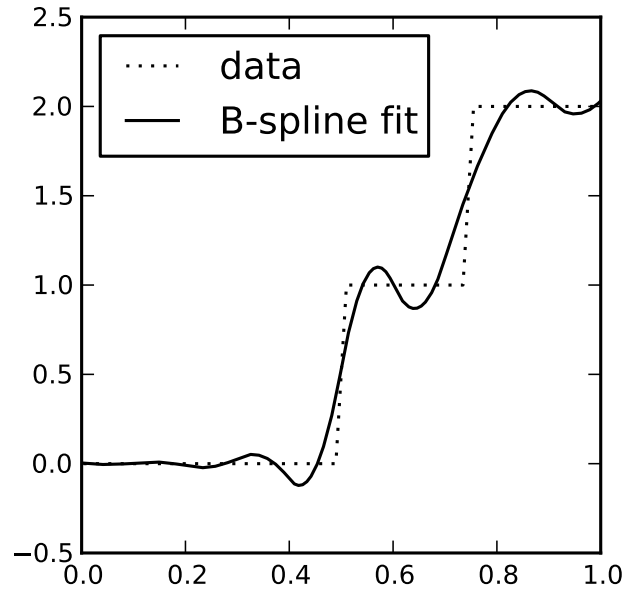


Figure 3: An example of the “ringing” that can occur as an artifact of least-squares fitting with B-splines. The sharp transitions are unrepresentable in the B-spline basis, so the spline surface must fluctuate around the data in order to minimize the residual. This can cause the spline surface to become negative where the data are strictly positive and non-monotonic where the data are strictly monotonic.

We implemented the method suggested by [13], in which we transform one axis of the tensor-product B-spline basis to a cumulative T-spline (“Trapezoidal” spline) basis in which each basis function is the sum of the B-spline basis functions following it. The equation for the coefficients of the penalized B-spline surface (5) can be converted to the T-spline basis by multiplying the basis and penalty matrices from the right with an upper-triangular matrix:

$$\tilde{B} = BU = B \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 0 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \quad (8)$$

Unlike the B-splines from which they are derived, the T-spline basis function have highly non-local support; e.g.

the first T-spline basis function has support over the entire knot field. This makes evaluating the spline surface arduous; in the extreme case of a point in the support of the last T-spline basis function, evaluating the spline surface involves all the preceding coefficients. Luckily, we can use the T-spline basis for fitting only, transforming the T-spline coefficients into the corresponding B-spline coefficients before storing them.

The T-spline basis, however, makes it particularly easy to enforce monotonicity in one dimension. In this basis, the spline surface is monotonic if and only if all the T-spline coefficients are positive. While not quite as simple as the unconstrained problem, there exist algorithms to solve the non-negative least-squares problem as a series of unconstrained problems in a finite number of steps.

Given  $A \in \mathbb{R}^{M \times N}$  and  $b \in \mathbb{R}^N$ ,  $x \in \mathbb{R}^M$  is the solution to the non-negative least squares problem:

$$\min_x \|Ax - b\|_2^2 : x \geq 0 \quad (9)$$

The Karush-Kuhn-Tucker optimality conditions lead to the following linear complementarity problem [14]:

$$y = A^T Ax - A^T b : y \geq 0, x \geq 0, x^T y = 0 \quad (10)$$

An active set approach is commonly used to find the solution to (10). The set of coefficients  $x$  is partitioned into a free set  $\mathcal{F}$  and a constrained set  $\mathcal{G}$  (often called the passive and active sets, respectively). The coefficients in the free set are the solution to the unconstrained least squares problem in the subspace of the free set, while those in the constrained set are clamped to zero. The problem then reduces to quickly and efficiently finding the partition of the coefficients that satisfies (10). The solution can be found iteratively, but unlike the minimization of a general function, it is possible to construct algorithms that are provably finite.

The best known of these is the Lawson-Hanson NNLS algorithm [15]. For very large problems, NNLS can be slow to converge, so we additionally implemented and examined two block-pivoting algorithms, the Portugal/Judice/Vincente (PJV) algorithm [14] and Adlers BLOCK3 [16], all using sparse matrix operations. Both provide substantial speed improvement relative to NNLS, but, due to occasional cycling behavior in the PJV algorithm, we recommend use of BLOCK3 for all applications. BLOCK3 provides rapid convergence even for very large problems (Figure 4, 5) and converges deterministically in all circumstances.

#### 4.2. Gradients of spline surfaces

Even though we fit the cumulative distribution of photon arrival times, we still need the probability density functions for sampling as well as the other dimensions of the gradient for accelerating minimizer convergence in maximum-likelihood fitting. Using the de Boor recursion relation [11, Chapter X] and the multiplication rule, we

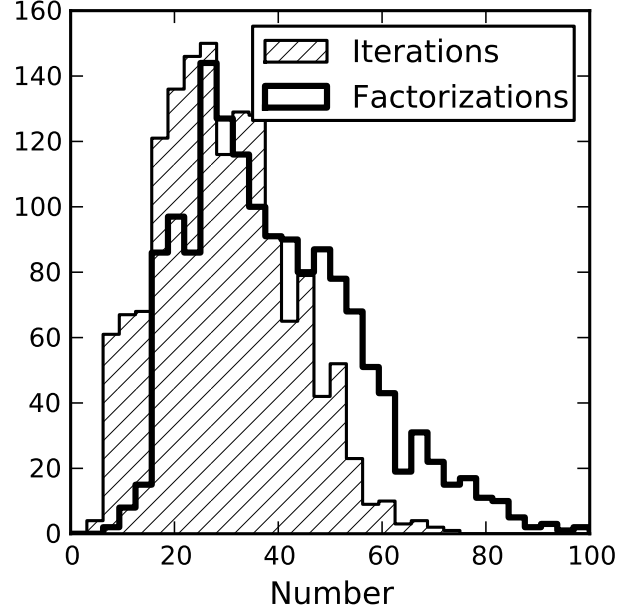


Figure 4: Number of iterations and complete Cholesky factorizations needed for the BLOCK3 algorithm to converge when fitting large multi-dimensional photon tables (100 million table cells and 400 thousand spline coefficient).

can easily construct a new set of basis functions that are the derivatives of B-splines:

$$B_n^{i'}(x) = \frac{(x - \vec{k}_i)B_{n-1}^{i'}(x) + B_{n-1}^i(x)}{\vec{k}_{i+n} - \vec{k}_i} + \quad (11)$$

$$\frac{(\vec{k}_{i+n+1} - x)B_{n-1}^{i+1'}(x) - B_{n-1}^{i+1}(x)}{\vec{k}_{i+n+1} - \vec{k}_{i+1}} \quad (12)$$

We can unpack one of these recursions in order to write the derivatives as simple linear combinations of two lower-order B-splines:

$$B_n^{i'}(x) = \frac{n}{\vec{k}_{i+n} - \vec{k}_i} B_{n-1}^i(x) - \frac{n}{\vec{k}_{i+n+1} - \vec{k}_{i+1}} B_{n-1}^{i+1}(x) \quad (13)$$

The lower-order basis functions needed to form the derivative basis can be obtained at almost no additional computational cost by pausing the bottom-up calculation of  $B_n^i$  [11, Algorithm BSPLVB] at order  $n-1$  before proceeding to the final step.

Because differentiation and addition commute, we can interchange these differential spline basis functions on the axis of interest with the original ones, while keeping all the other axes and the coefficient table unchanged. Keeping the coefficients unchanged also allows us to efficiently evaluate the spline surface and its gradient using SIMD operations. In our case, evaluating the spline surface and the 6 elements of its gradient in sequence takes 7 times as long

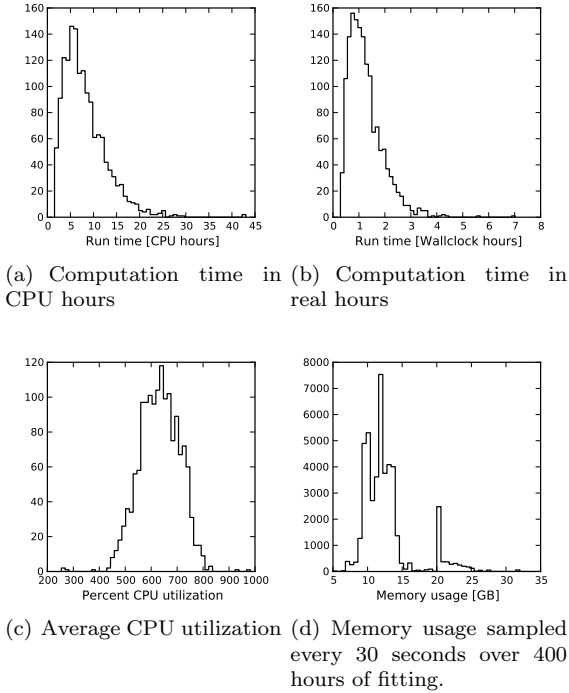


Figure 5: Resource consumption of penalized spline fits. Shown are two extremely large fitting jobs (100 million table cells and 400 thousand coefficients) running in parallel on a 16-core, 2.3 GHz AMD Opteron system with 64 GB of memory using SuiteSparse [9] and GotoBLAS 2 [10]. The memory usage peak at 20 GB represents the memory needed for the initial factorization of the full  $A^T A$ , while the broader, lower-memory peak represents the memory needed for the factorization of only the unconstrained parts of  $A^T A$ .

as a single evaluation, while the same computation implemented in terms of operations on 4-element basis vectors takes slightly less than twice as long.

#### 4.3. Analytic convolution of spline surfaces

While scattering processes account for the vast majority of the time-delay distribution in detected photons, it can be important to account for detector effects as well when reconstructing events. In particular, the propagation of current pulses through the photomultiplier tube and inter-DOM clock synchronization [2] introduce small timing uncertainties in the measured photon arrival time distribution. Also, while electromagnetic and hadronic showers are approximately point-like on the scale of the inter-string spacing in IceCube, they do have some spatial extent. Both of these effects can be approximately accounted for by convolving the photon delay-time distribution with a Gaussian.

On a uniform knot field, a B-spline basis function of order  $n$  is proportional to the  $n$ -fold self-convolution of a constant function between two adjacent knots, and so the  $n$ th order uniform B-spline becomes proportional to a normal distribution as  $n \rightarrow \infty$ . B-splines of relatively low order can be used to approximate Gaussian convolution kernels, albeit with limited support. Furthermore, it is possible to analytically convolve a spline surface with a B-spline kernel and expand the result in a new B-spline basis (Fig. 6). These are useful properties, as they allow convolution of a spline surface by a pseudo-Gaussian kernel merely by manipulating its coefficients.

The procedure for analytically convolving splines is described in [8] in a slightly different form than we require for our application, and we will review our adaptation of it here for the sake of clarity. As before, we will use  $B_m^i$  to denote the  $i$ th B-spline of order  $m$  on a knot vector  $\vec{t}$  with the usual de Boor normalization such that the set of splines on  $\vec{t}$  sum to 1 everywhere. The convolution kernel will be represented by a spline  $M_n^j$  on a knot vector  $\vec{\tau}$ , but normalized so that each individual basis function integrates to 1.

The convolution of  $B_m^i$  and  $M_n^j$  can be written<sup>1</sup> in terms of dummy variables  $x$  and  $y$  as

$$\begin{aligned} (B_m^i \star M_n^j)(z) &= (t_{i+m+1} - t_i) \frac{n!(m+1)!}{(m+n+1)!} \\ &\quad \times [t_i, \dots, t_{i+m+1}]_x [\tau_j, \dots, \tau_{j+n+1}]_y \\ &\quad \times (x + y - z)_+^{m+n+1} \end{aligned} \quad (14)$$

where  $[t_i, \dots, t_{i+m+1}]_x$  is the divided-difference operator with respect to  $x$  and  $(\cdot - z)_+^n$  is the truncated power function

$$(a - b)_+^n = \begin{cases} (a - b)^n & b < a \\ 0 & b \geq a \end{cases} \quad (15)$$

<sup>1</sup>This is a variation on Equation 13 of [8].

Furthermore,  $B_m^i \star M_n^j$  is a B-spline surface of order  $m+n+1$  on the combined knot field  $\vec{\rho} \equiv \{t_i, \dots, t_{i+m+1}\} \uplus \{\tau_j, \dots, \tau_{j+n+1}\}$ <sup>2</sup>. Given this fact and the ability to evaluate  $(B_m^i \star M_n^j)(z)$  for arbitrary  $z$ , we can expand the convolution in its B-spline coefficients.

A B-spline surface of order  $m$  defined on a knot field  $\vec{\rho}$  is a linear combination of B-splines:

$$f(x) = \sum_i a_i B_{m,\vec{\rho}}^i \quad (16)$$

The coefficients of the B-spline expansion are given by

$$a_i = \mathcal{B}(f(\xi_i))(\rho_{i+1}, \dots, \rho_{i+m}) \quad (17)$$

where  $\xi_i \in [\rho_i, \rho_{i+m})$  and  $\mathcal{B}$  is the multilinear blossom of  $f$  [17]. The multilinear blossom is a function that is linear in each of its  $m$  arguments and is identical to  $f$  when evaluated on its diagonal, that is, when all its arguments are identical. The blossom of  $B_m^i \star M_n^j$  is given by

$$\begin{aligned} \mathcal{B}((B_m^i \star M_n^j)(z))(\rho_{i+1}, \dots, \rho_{i+m+n+1}) &= (t_{i+m+1} - t_i) \\ &\times \frac{n!(m+1)!}{(m+n+1)!} [t_i, \dots, t_{i+m+1}]x[\tau_j, \dots, \tau_{j+n+1}]y \\ &\times \Theta(x+y-z) \prod_{l=i+1}^{i+m+n+1} (x+y-\rho_l) \end{aligned} \quad (18)$$

where  $\Theta$  is the Heaviside step function.

In our application the convolution kernel is given by a single unit-normalized spline  $M_n^1$ , and we can apply (17) to write the coefficients  $\vec{a}$  of the B-spline expansion of the convolved surface in terms of the coefficients  $\vec{b}$  of the unconvolved surface as

$$\vec{a} = T\vec{b} \quad (19)$$

where  $T$  is a matrix whose elements can be found by evaluating the blossom defined in (18):

$$T_{ij} = \mathcal{B}((B_m^j \star M_n^1)(\rho_i))(\rho_{i+1}, \dots, \rho_{i+m+n+1}) \quad (20)$$

This is particularly useful for tensor-product spline surfaces: the convolution can be carried out along a given dimension by calculating the transformation matrix once and then applying it to each slice of the coefficient grid along that dimension.

<sup>2</sup>The combined knot field is formed by  $m+2$  copies of  $\{\tau_j, \dots, \tau_{j+n+1}\}$ , each centered on an element of  $\{t_i, \dots, t_{i+m+1}\}$ . For example,  $\{0, 1, 2\} \uplus \{-0.1, 0, 0.1\} = \{-0.1, 0, 0.1, 0.9, 1, 1.1, 1.9, 2, 2.1\}$ . For a proof that  $B_m^i \star M_n^j$  is contained in the spline space of order  $m+n+1$ , see Theorem 8 of [8].

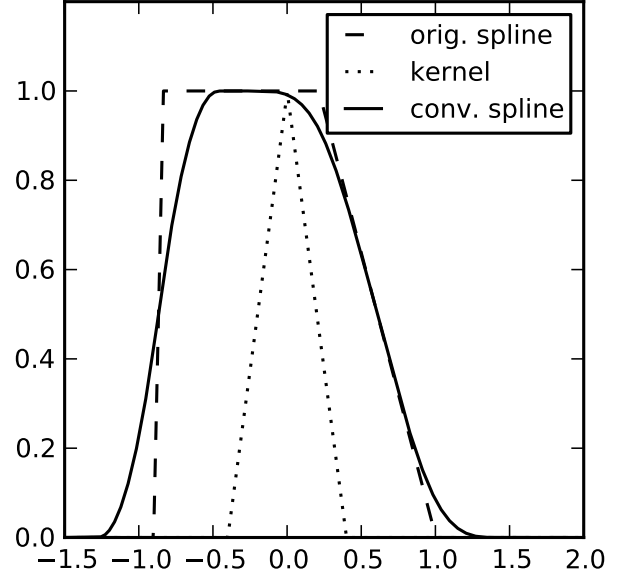


Figure 6: An example of spline convolution. Here, an order-1 spline surface on a non-uniform knot field is convolved with an order-1 spline, resulting in an order-3 spline surface.

## 5. Conclusion

Penalized splines provide a number of attractive properties for handling detector-response simulations in high-energy physics experiments where alternative parametrizations are not practicable: fast and efficient deterministic fitting, even for very large datasets, integrated smoothing and extrapolation, and the ability to easily perform a variety of mathematical operations. The conventional penalized spline technique [1] was extended here with new tools and with existing algorithms [8, 15, 16] for convolution of the resulting spline tables and use with datasets typical in high-energy physics. Use of multi-dimensional tabulated Monte Carlo data in maximum-likelihood fits can be critical for many complicated situations such as detailed particle interaction reconstruction in non-segmented inhomogeneous detectors such as IceCube [5]. Most other tabulation strategies (multilinear interpolation, polynomials) quickly become intractable as the dimensionality of the problem increases due to numerical issues or computational footprint, rendering problems that can be easily solved in principle impossible or extremely difficult in practice. Use of spline surfaces, along with the ability to perform on-the-fly differentiation, integration, and convolution, solves many of these problems and greatly improves the ability to use such tables in maximum likelihood fits. Similar problems of detector response parametrization occur across high-energy physics; use of penalized splines may help with related problems in many detectors.

## Acknowledgments

We would like to thank Brian Marx and Paul Eilers for invaluable assistance with implementation of the fitting algorithms. J. van Santen was supported under NSF grant 0636875 and N. Whitehorn by the NSF GRFP. Computing resources provided with support from the NSF Office of Polar Programs and Physics Division as well as from the U. of Wisconsin Alumni Research Foundation. Thanks to S. BenZvi for packaging the fitting software for external use.

## References

- [1] B. Marx, P. Eilers, Flexible smoothing with B-splines and penalties (with comments and rejoinder), *Statistical Science* 11 (2) (1996) 89–121. doi:10.1214/ss/1038425655.
- [2] R. Abbasi, et al., The IceCube data acquisition system: Signal capture, digitization, and timestamping, *Nucl. Instrum. and Meth. A* 601 (3) (2009) 294 – 316. doi:10.1016/j.nima.2009.01.001.
- [3] J. Ahrens, et al., Muon track reconstruction and data selection techniques in AMANDA, *Nucl. Instrum. and Meth. A* 524 (2004) 169–194. doi:10.1016/j.nima.2004.01.065.
- [4] S. Grullon, D. J. Boersma, G. Hill, K. Hoshina, K. Mase, Reconstruction of high energy muon events in IceCube using waveforms, in: *Proceedings of the 30th International Cosmic Ray Conference*, Merida, Yucatan, Mexico, 2007, pp. 1457–1460. arXiv:0711.0353.
- [5] E. Middell, J. McCartin, M. D’Agostino, Improved reconstruction of cascade-like events in IceCube, in: *Proceedings of the 31st International Cosmic Ray Conference*, Łódź, Poland, 2009, p. 708.
- [6] M. Ackermann, et al., Optical properties of deep glacial ice at the South Pole, *J. Geophys. Res.* 111 (D13) (2006) 1–26. doi:10.1029/2005JD006687.
- [7] P. H. C. Eilers, I. D. Currie, M. Durbán, Fast and compact smoothing on large multidimensional grids, *Computational Statistics and Data Analysis* 50 (2006) 61–76. doi:10.1016/j.csda.2004.07.008.
- [8] K. Strøm, On convolutions of B-splines, *Journal of Computational and Applied Mathematics* 55 (1) (1994) 1 – 29. doi:10.1016/0377-0427(94)90182-1.
- [9] Y. Chen, T. A. Davis, W. W. Hager, S. Rajamanickam, Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate, *ACM Transactions on Mathematical Software* 35 (3) (2008) 22:1–22:14. doi:10.1145/1391989.1391995.
- [10] K. Goto, R. A. van de Geijn, Anatomy of high-performance matrix multiplication, *ACM Transactions on Mathematical Software* 34 (3) (2008) 12:1–12:25. doi:10.1145/1356052.1356053.
- [11] C. de Boor, *A Practical Guide to Splines*, Vol. 27 of *Applied Mathematical Sciences*, Springer-Verlag, 1978.
- [12] J. Lundberg, P. Miočinović, K. Woschnagg, T. Burgess, J. Adams, S. Hundertmark, P. Desiati, P. Niessen, Light tracking through ice and water—scattering and absorption in heterogeneous media with Photonics, *Nucl. Instrum. and Meth. A* 581 (3) (2007) 619 – 631. arXiv:astro-ph/0702108v2, doi:10.1016/j.nima.2007.07.143.
- [13] G. Beliakov, Shape preserving approximation using least squares splines, *Approximation Theory and its Applications* 16 (4) (2000) 80–98. doi:10.1007/BF02837633.
- [14] L. F. Portugal, J. J. Judice, L. N. Vicente, A comparison of block pivoting and interior-point algorithms for linear least squares problems with nonnegative variables, *Mathematics of Computation* 63 (208) (1994) 625–643. doi:10.2307/2153286.
- [15] C. L. Lawson, R. J. Hanson, *Solving least squares problems*, Prentice-Hall, Englewood Cliffs, NJ, 1974.
- [16] M. Adlers, Sparse least squares problems with box constraints, Ph.D. thesis, Linköpings Universitet, S-581 83 Linköping, Sweden (1998). URL <http://www.mai.liu.se/~milun/lic.ps>
- [17] L. Ramshaw, Blossoming: A connect-the-dots approach to splines, Tech. Rep. 19, Digital Equipment Corporation (1987). URL <http://www.hpl.hp.com/techreports/Compaq-DEC/SRC-RR-19.pdf>